Particle Swarm Optimisation

Group 3

Contents

1 Particle Swarm Algorithm

1.1 Background and Algorithm

The particle swarm optimization (PSO) algorithm was first developed by James Kennedy and Russell Eberhart in 1995, who were inspired by the movement of schools of fish and flocks of birds. Computer simulations of the flocking behavior of animals are common in the field of artificial life and are often used for computer animations (Hu). Kennedy and Russell observed

that in a group, these animals were able to utilize both their own knowledge as well as the group's knowledge in order to navigate. The particle swarm optimization method uses this idea by iteratively moving particles across the search space toward a weighted average of their own best position and the group's best position (Eberhart & Kennedy, 1995). The algorithm is given below (Poli, Kennedy, & Blackwell, 2007):

Algorithm 1

1. Initialize an array of N d-dimensional particles with random positions $\{x^i\}_{i=1}^N$ and velocities $\{v^i\}_{i=1}^N$ in the search space (i is an index here, not an exponent).
2. Repeat until a convergence criterion is met or the maximum number of iterations ($k = 1, \ldots, k_{max}$) is reached:
    1. For each particle, evaluate the fitness function (the objective function), $f(x_k^i)$. Compare the current fitness value with the particle's best fitness value found so far, $pbest_{k-1}^i$. If the current value is better, set $pbest_k^i$ equal to the current value and the particle's best position, p_k^i to the current position.
    2. Find the particle with the best fitness value and set the neighbourhood's best fitness value $pbest_k^g$ and best position p_k^g to be the fitness value and position of this particle respectively.
    3. For each particle, update the position and velocity of particle i at iteration $k+1$ according to the equations:

$$x_{k+1}^i = x_k^i + v_{k+1}^i \tag{1}$$

$$v_{k+1}^i = w_k v_k^i + c_1 r_1 (p_k^i - x_k^i) + c_2 r_2 (p_k^g - x_k^i) \tag{2}$$

    where w_k is an inerta parameter, c_1 is a cognitive parameter, c_2 is a social parameter, and r_1 and r_2 are random numbers uniformly distributed in $[0, 1]$.

The algorithm starts out with particles moving randomly around the search space, then in each iteration, takes note of each particle's personal best position visited so far and the best position found in the group, updating the each particle' movement according to (i) the particle's previous direction, weighted by w, (ii) the direction towards personal best position (self-cognitive behaviour), weighted by $\phi_1 = c_1 r_1$, and (iii) the direction towards the group's best position (social behaviour), weighted by $\phi_2 = c_2 r_2$. The general role

of the model parameters is to create a balance between exploration and exploitation, which determines the algorithm performance.

"Exploration" refers to the ability of the algorithm to test many different parts of the search space, while "exploitation" is the precision with which the algorithm is able to concentrate around an optimum (Trelea, 2003). A higher level of exploration means the algorithm is less likely to converge to a local optimum rather than a global one, while a higher level of exploitation means that the algorithm will converge more quickly, but it might be to a local optimum rather than a global one.

1.1.1 Modifications and Variants

A few major modifications to PSO are discussed in this section since they give us a better understanding of the design and the parameters used in PSO. The original PSO model does not have the inertia parameter and uses $c_1 = c_2 = 2$, however this causes to "explode", that is, the particles' trajectories are divergent and many will even move out of the search space. This will be proven in section 1.2. Hence, "velocity clamping" is introduced by setting bounds $[V_{min}, V_{max}]$ for particles' velocities. However, this cannot ensure the convergence of PSO.

Shi & Eberhart (1998) modify the PSO by adding the inertia parameter, as a means to balance the exploration and exploitation of the model, and at the same time eliminate the need for velocity clamping. The inertia parameter, w, effects how easily the particles will move through the search space. Poli et al. (2007) describe it as the "fluidity of the medium in which a particle moves". w typically takes values between 0 and 1, and when it is close to zero, the exploration of the algorithm will be low. On the other hand, when w is close to 1, exploration will be high, with particles moving through "low viscosity medium". One strategy that is commonly employed to balance these effects is starting the algorithm with a high inertia coefficient and gradually reducing it toward zero as the iterations progress.

To remove the need for velocity clamping and to guarantee convergence, Clerc & Kennedy (2002) modify the original PSO by adding a constriction coefficient χ to the velocity update equation as showed below:

$$v_{k+1}^i = \chi \left(v_k^i + c_1 r_1 (p_k^i - x_k^i) + c_2 r_2 (p_k^g - x_k^i) \right)$$

χ is derived theoretically such that convergence is ensured, however, it is not straightforward to derive from this model a practical guideline for parameters selection (Trelea, 2003).

In addition to these two important modifications, different topologies have been considered (Poli et al., 2007) and many variants of PSO have been developed for different purposes (Poli et al., 2007, p. @cheng2018quarter).

In the rest of the report, PSO model with inertia parameter, as showed in **Algorithm 1**, and with the neighbourhood being the entire swarm (the so-called "gbest" topology) will be used.

1.2 Convergence Analysis

In contrast to the normal modelling problems in which models are designed to fit the observed natural occurrences, we have an exact model of PSO which is complicated and thus we want to approximate this model with a simplified one on which convergence analysis is viable. There are primarily two approaches: deterministic model (where the stochastic components in PSO are assumed to be fixed) and stochastic model (where the previous assumption is dropped). The focus of this section is on the deterministic model for the PSO in **Algorithm 1** as it leads us to practical criteria on the model parameters w, $\phi_1 = c_1 r_1$ and $\phi_2 = c_2 r_2$.

1.2.1 Deterministic PSO Model

We will first examine the deterministic model presented in the works of Trelea (2003) and Van den Bergh & Engelbrecht (2006). Besides the *deterministic assumption* that ϕ_1 and ϕ_2 are fixed, a further assumption, coined as *stagnation assumption* by Cleghorn & Engelbrecht (2014), that $p_k^i = p^i$ and $p_k^g = p^g$ for all k, is made. Note that this assumption still allows different particles to have different (but fixed) personal best positions. Without loss of generality, the convergence analysis will be concentrated on a one-dimensional particle (so that we can drop the index i), and it will be clear that the result applies to every particle and to a higher dimensional case. It will be showed that the particle converges to a stable point, which will be a weighted average of p^i and p^g. Note that whether this point is a local or global optimum is not implied from this analysis, and what is crucial here is that each particle will converge to a stable point and the swarm attains equilibrium.

Consider the position and velocity update equations, Eq.(1) and Eq.(2) in **Algotrithm 1**, but with $c_1 r_1$ and $c_2 r_2$ replaced with ϕ_1 and ϕ_2, the particle index i dropped, and with p_k^i and p_k^g set to fixed values p and p^g:

$$x_{k+1} = x_k + v_{k+1} \tag{3}$$

$$v_{k+1} = w_k v_k + \phi_1(p - x_k) + \phi_2(p^g - x_k) \tag{4}$$

Substituting Eq.(4) into Eq.(3) gives the following recurrence relation between x_{k+1} and its previous two positions x_k and x_{k-1}:

$$x_{k+1} = (1 + w - \phi_1 - \phi_2)x_k - wx_{k-1} + \phi_1 p + \phi_2 p^g \tag{5}$$

which can be written in the following matrix form:

$$\begin{bmatrix} x_{k+1} \\ x_k \end{bmatrix} = A \begin{bmatrix} x_k \\ x_{k-1} \end{bmatrix} + b$$

where

$$A = \begin{bmatrix} 1 + w - \phi_1 - \phi_2 & -w \\ 1 & 0 \end{bmatrix} \quad b = \begin{bmatrix} \phi_1 p + \phi_2 p^g \\ 0 \end{bmatrix}$$

By solving the characteristic polynomial of matrix A:

$$\lambda^2 - (1 + w - \phi_1 - \phi_2)\lambda + w = 0$$

the eigenvalues can be obtained in terms of the fixed model parameters:

$$\lambda_1 = \frac{(1 + w - \phi_1 - \phi_2) + \gamma}{2}$$

$$\lambda_2 = \frac{(1 + w - \phi_1 - \phi_2) - \gamma}{2}$$

where $\gamma = \sqrt{(1 + w - \phi_1 - \phi_2)^2 - 4w}$, which is complex when $(1 + w - \phi_1 - \phi_2)^2 < 4w$.

The explicit closed form solution of the recurrence Eq.(5), in terms of x_0 and x_1, is then:

$$x_k = \beta_1 + \beta_2 \lambda_1^k + \beta_3 \lambda_2^k \tag{6}$$

where

$$\beta_1 = \frac{\phi_1 p + \phi_2 p^g}{\phi_1 + \phi_2}$$
$$\beta_2 = \frac{\lambda_2(x_0 - x_1) - x_1 + x_2}{\gamma(\lambda_1 - 1)}$$
$$\beta_3 = \frac{\lambda_1(x_1 - x_0) + x_1 - x_2}{\gamma(\lambda_2 - 1)}$$
$$x_2 = (1 + w - \phi_1 - \phi_2)x_1 - wx_0 + \phi_1 p + \phi_2 p^g$$

For a complex eigenvalue λ,

$$\lim_{k\to\infty} \lambda^k = \lim_{k\to\infty} ||\lambda||^k e^{i\theta k} = \lim_{k\to\infty} ||\lambda||^k (\cos\theta k + i \sin\theta k)$$

where $\theta = arg(\lambda)$. Hence, $\lim_{k\to\infty} \lambda^k = 0$ if $||\lambda|| < 1$, and this also holds for real eigenvalue. The explicit solution in Eq.(6) thus implies that if $\max\{||\lambda_1||, ||\lambda_2||\} < 1$, then

$$\lim_{k\to\infty} x_k = \beta_1 = \frac{\phi_1 p + \phi_2 p^g}{\phi_1 + \phi_2}$$

At this point, we will drop the *deterministic assumption* and take into account the randomness of ϕ_1 and ϕ_2. Since r_1 and r_2 are uniformly distributed in $[0, 1]$, taking expectation of β_1 gives

$$\begin{aligned}
\mathbb{E}(\beta_1) &= \frac{\frac{c_1}{2} p + \frac{c_2}{2} p^g}{\frac{c_1}{2} + \frac{c_2}{2}} = \frac{c_1 p + c_2 p^g}{c_1 + c_2} \\
&= \frac{c_1}{c_1 + c_2} p + \frac{c_2}{c_1 + c_2} p^g = \frac{c_1}{c_1 + c_2} p + \left(1 - \frac{c_1}{c_1 + c_2}\right) p^g \\
&= \alpha p + (1 - \alpha) p^g
\end{aligned}$$

where $\alpha = \frac{c_1}{c_1 + c_2} \in [0, 1]$.

Hence, under the condition that $\max\{||\lambda_1||, ||\lambda_2||\} < 1$, it has been showed that the particle converges to a stable point which is, on average, the weighted average of its best position and the global best position.

1.2.2 Generalised Deterministic Model and Stochastic Model

However, the *stagnation assumption* is unrealistic, at least for two reasons (Cleghorn & Engelbrecht, 2014). First, PSO relies on the social interaction between particles which becomes meaningless if p^i and p^g are fixed. Second, when PSO is stagnated, then it is no longer an optimiser as subsequent iterations will not give better solution. Therefore, Cleghorn & Engelbrecht (2014) replace the *stagnation assumption* with the weaker assumption that p^i and p^g can take an arbitrarily large but finite number of distinct values, and then prove, under the same framework as laid out above, that each particle converges to a point if $\max\{||\lambda_1||, ||\lambda_2||\} < 1$. This results in a generalised deterministic model, which yields the same result, with the same condition, but with a weaker assumption.

In addition, several stochastic models have been formulated to approximate the PSO model (refer to (Cleghorn & Engelbrecht, 2014) for a brief coverage of these models). For a more recent work, the swarm state sequence, in which the state of a particle i at iteration k is defined as $(x_{k-1}, x_k, p_k^i, p_k^g)$, is modelled as a Markov chain and then the convergence analysis is carried out by utilising supermartingale convergence theorem (Xu & Yu, 2018).

1.2.3 Parameter Condition for Convergence

The condition of $\max\{||\lambda_1||, ||\lambda_2||\} < 1$ for the convergence to a stable point implies, the following set of conditions on the model parameters which can be employed as a practical guideline for parameters selection:

$$||w|| < 1 \quad 0 < \phi_1 + \phi_2 < 4 \quad w > \frac{\phi_1 + \phi_2}{2} - 1$$

The intersection area of these constraints is showed as area A in the plot below:

Notice that there are still stochastic components ϕ_1 and ϕ_2 in the conditions, as these are the general conditions for the generalised deterministic model (Cleghorn & Engelbrecht, 2014). If we replace the random $\phi_1 + \phi_2$ with their expected value, which is just $\frac{c_1+c_2}{2}$, then we arrive at the conditions for convergence presented in Trelea (2003), with $b = \frac{c_1+c_2}{2}$:

$$-1 < w < 1 \quad b > 0 \quad w > \frac{b}{2} - 1 \quad (w > \frac{c_1 + c_2}{4} - 1)$$

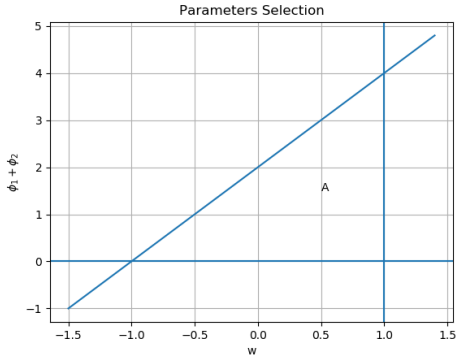

Figure 1: 'Figure 1'

Instead of using the expected value, if we replace the randomness with the upper bound of $\phi_1 + \phi_2$, which is $c_1 + c_2$ since $r_1, r_2 \in [0, 1]$, then we obtain the conditions of convergence delineated in Van den Bergh & Engelbrecht (2006):

$$0 < w < 1 \quad c_1 > 0 \quad c_2 > 0 \quad w > \frac{c_1 + c_2}{2} - 1$$

Comparing the last inequality in the two conditions, convergence condition of Van den Bergh & Engelbrecht is more restrictive than Trelea's condition. More specifically, the use of upper bounds c_1 and c_2 in the condition of Van den Bergh & Engelbrecht encourages convergence in every iteration while the use of expected values in Trelea's condition ensures that roughly 50% of all the iterations will encourage convergence.

Now we can show that the original PSO model without the inertia weight and with $c_1 = c_2 = 2$ will have divergent particle trajectory. When there is no inertia weight, $w = 1$, and let $\phi = \phi_1 + \phi_2$, then since $c_1 = c_2 = 2$, $\phi \in [0, 4]$. The eigenvalues will be complex since $(2 - \phi)^2 \leq 4$ and

$$\gamma = \sqrt{(2-\phi)^2 - 4} = \sqrt{\phi^2 - 4\phi} = i\sqrt{4\phi - \phi^2}$$

The eigenvalues are complex conjugates, and hence they have the same modulus:

$$||\lambda_1|| = ||\lambda_2|| = \sqrt{\frac{(2-\phi)^2}{4} + \frac{4\phi - \phi^2}{4}} = 1$$

resulting in the violation of the condition $\max\{||\lambda_1||, ||\lambda_2||\} < 1$.

1.2.4 Parameters Selection Hueristic

With the parameter condition in the last section, we are still left many choices of parameters. Trelea (2003) suggest the following user-oriented selection hueristic.

Notice first that there are two other factors that influence the algorithm performance:

1. The trade-off between exploration and exploitation depends largely on the function being optimized, particularly the number of optimums and their locations. Hence, it might not be possible to derive a set of parameters which cater for all functions.

2. The number of particles used is one important parameter. A large number of particles means that the function will need to be evaluated many times during each iteration, which could slow down the algorithm, but using too few particles will cause the algorithm to converge more slowly. Typically, 20-50 particles are used, depending on the difficulty of the optimization problem (Poli et al., 2007).

With these two factors in mind, Trelea suggests that the best method for choosing parameters is to start with those that will allow the algorithm to converge quickly (eg. larger values of model parameters but still within the triangle area A), and if it gives different solutions each time it is run (convergence rate is too fast, and particles converge prematurely to local optimums), use parameters that will slow the convergence until the same result can be obtained consistently (eg. larger number of particles). In addition to this manual, a popular choice of parameter set is that $w = 0.7298$ and $c_1 = c_2 = 1.49618$, which has been empirically proven to produce good results on benchmark test functions.

1.3 Applications

PSO algorithm could be used widely in both academic and real-life problem.

1.3.1 Academic Problems

For back propagation neural network in deep learning, the weight in multi-layers as well as the thresholds could be trained by PSO algorithm instead

of the traditional gradient decent method and in such way, it could improve error and iteration time for BP network (Hu & Song, 2014). Another application of PSO in data mining is to clustering high-dimension data. Compared to the K-means clustering algorithm, PSO could compute more dense clusters (Esmin, Coelho, & Matwin, 2015). There are several slightly different clustering algorithms based on PSO, including Alternative KPSO clustering (AKPSO), mountain clustering based on an improved PSO(MCBIPSO), dynamic clustering based on PSO etc. The strengths for embracing PSO into clustering algorithm could be identifying the centres of clusters more efficiently; determining the density and the number of clusters more appropriately; or escaping from a local optimum. In addition, PSO could also be applied for classic operations research problem – traveling salesman problem. According to Pang et al. (2004), by mapping from continuous space to permutation space by implementing PSO, the traveling salesman problem could be solved effectively.

1.3.2 Real-life Applications

For real-life application, PSO could be used in wireless sensor networks to reduce the computational complexity (Arastu, 2012). PSO could improve the accuracy and the speed of convergence when solving sintering blending for the cost reduction (Zhao, Wang, Wang, & Yue, 2012). Parallel synchronous PSO (PS-PSO) can be applied on line power system studies as a heuristic algorithm finding the sub-optimums but significantly saving computational costs (Abderrahmani, Nasri, & DEHINI, 2011).

2 Parallelisation

References

Abderrahmani, A., Nasri, M., & DEHINI, R. (2011). Economic emission dispatch solution using parallel synchronous pso algorithms. *Acta Electrotechnica et Informatica*, *11*(1), 66–70.

Arastu, S. H. (2012). *Distributed particle swarm optimizer for wireless sensor networks* (PhD thesis). *ProQuest Dissertations and Theses*. Retrieved from http://ezproxy.library.usyd.edu.au/login?url=https://search-proquest-com.ezproxy1.library.usyd.edu.au/docview/1313228896?accountid=14757

Cheng, S., Lu, H., Lei, X., & Shi, Y. (2018). A quarter century of particle swarm optimization. *Complex & Intelligent Systems*, *4*(3), 227–239.

Cleghorn, C. W., & Engelbrecht, A. P. (2014). A generalized theoretical deterministic particle swarm model. *Swarm Intelligence*, *8*(1), 35–59.

Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, *6*(1), 58–73.

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the sixth international symposium on micro machine and human science* (pp. 39–43). Ieee.

Esmin, A. A., Coelho, R. A., & Matwin, S. (2015). A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. *Artificial Intelligence Review*, *44*(1), 23–45.

Hu, P., & Song, X. Q. (2014). On pso based bp neural network. *Applied Mechanics and Materials*, *602-605*, 3518–3521. Retrieved from http://ez proxy.library.usyd.edu.au/login?url=https://search-proquest-com.ezproxy1 .library.usyd.edu.au/docview/1826072335?accountid=14757

Pang, W., Wang, K.-P., Zhou, C.-G., Dong, L.-J., Liu, M., Zhang, H.-Y., & Wang, J.-Y. (2004). Modified particle swarm optimization based on space transformation for solving traveling salesman problem. In *Proceedings of 2004 international conference on machine learning and cybernetics (ieee cat. No. 04EX826)* (Vol. 4, pp. 2342–2346). IEEE.

Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization. *Swarm Intelligence*, *1*(1), 33–57.

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. In *1998 ieee international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (cat. No. 98TH8360)* (pp. 69–73). IEEE.

Trelea, I. C. (2003). The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters*, *85*(6), 317–325.

Van den Bergh, F., & Engelbrecht, A. P. (2006). A study of particle swarm optimization particle trajectories. *Information Sciences*, *176*(8), 937–971.

Xu, G., & Yu, G. (2018). Reprint of: On convergence analysis of particle swarm optimization algorithm. *Journal of Computational and Applied Mathematics*, *340*, 709–717.

Zhao, H., Wang, M., Wang, H. J., & Yue, Y. J. (2012). Study of sintering blending based on swarm intelligence optimization algorithm. In *Applied mechanics and materials* (Vol. 198, pp. 1550–1553). Trans Tech Publ.